Министерство образования и науки Российской Федерации

**Санкт-Петербургский политехнический университет Петра Великого**

**Институт компьютерных наук и технологий**

**Высшая школа киберфизических систем и управления**

Приложение для анализа динамических звеньев

**По курсу: Проектная работа**

Группа 3532704/00501

Студент: Кривов Г.М.

Студент: Александров А.В.

Утвердил Милицын А.В.

Санкт-Петербург

2023

Техническое задание на разработку настольного приложения для анализа динамических звеньев

**Терминология**

Под динамическим звеном понимается САУ, которая может быть представлена в виде передаточной функции в форме Боде (формула 1).

**Цель создания приложения**

Цель создания приложения – обеспечить пользователям простой инструментарий для анализа звеньев с помощью задания его передаточной функции.

**Требования к приложению**

**Приложение должно по полученной от пользователя функции получать четыре характеристики звена:**

1. **Переходную характеристику;**
2. **Амплитудно-частотную характеристику**
3. **Фазово-частотную характеристику;**
4. **Корневую картину.**

Переходная характеристика(h(t)) – реакция звена на подачу на его вход единичного воздействия(1(t)).

Амплитудно-частотная характеристика – зависимость отношения амплитуд входного и выходного сигналов от частоты.

Фазово-частотная характеристика – зависимость фазового сдвига выходного сигнала относительно входного от частоты сигнала.

Корневая картина – изображения корней переходной функции на комплексной плоскости. Корни числителя называются нулями, а корни знаменателя – полюсами.

**Приложение должно быть кроссплатформенным.**

С целью обеспечения данного критерия приложение будет написано с использованием Qt. Также с учетом использования математических вычислений программа будет реализована на языке программирования Python.

**Целевая аудитория**

Приложение нацелено на студентов и инженеров АСУ ТП, для быстрого и легкого получения характеристик звеньев.

**Макет приложения**

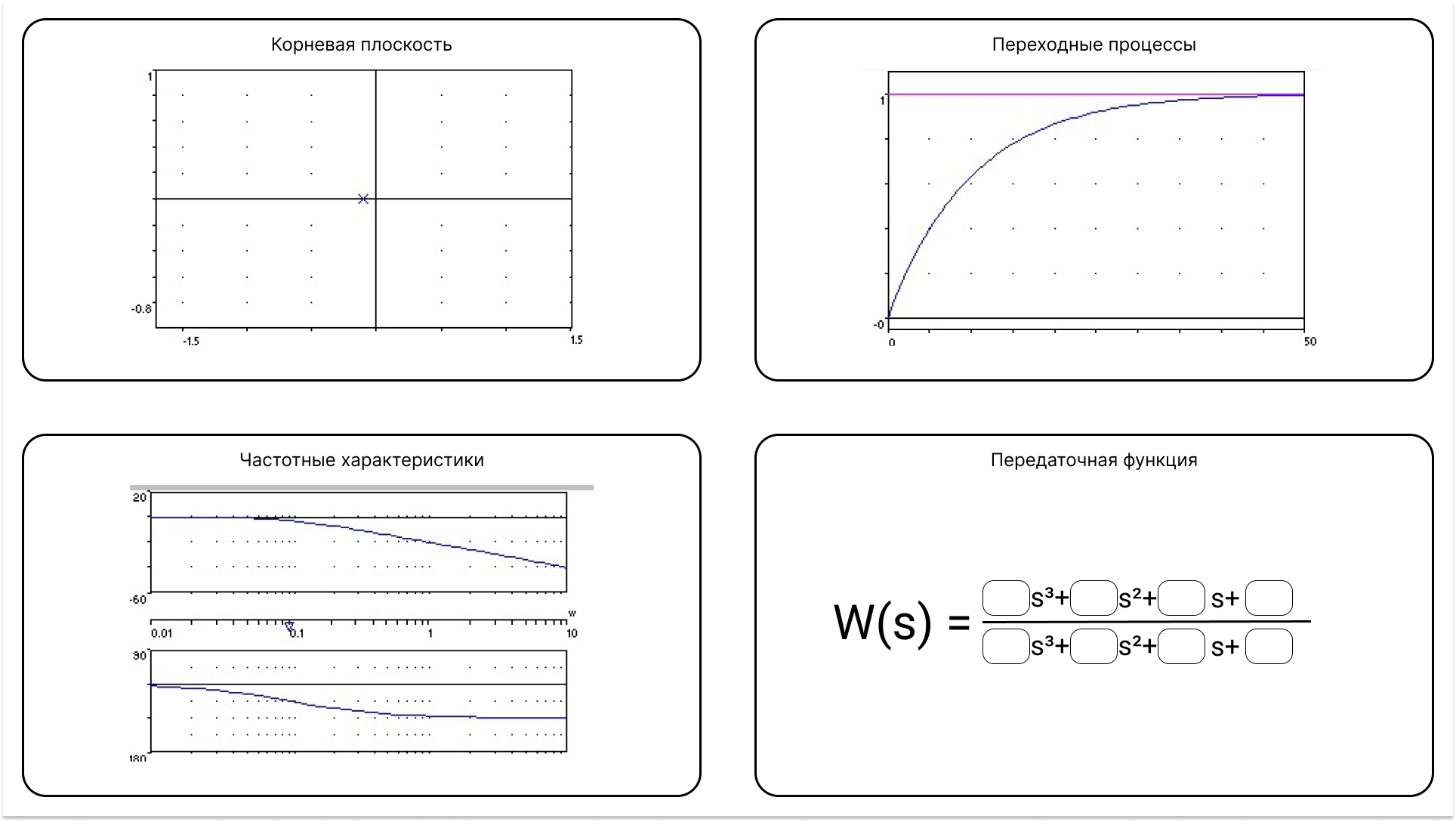


Рисунок 1. Макет приложения

План разработки приложения для анализа динамических звеньев

# 1 этап. Реализация математических функций, используемых для анализа звена, на выбранном языке программирования.

Срок выполнения этапа – 4 недели. На данном этапе необходимо создать функции для вычисления характеристик звеньев, которые в последствие будут интегрированы в пользовательский интерфейс. Так как для разработки был выбран язык программирования Python, то решение задач получения функции и построения графика облегчено обширным набором библиотек.

# 2 этап. Разработка пользовательского интерфейса

Срок выполнения этапа – 2 недели. На данном этапе необходимо разработать интерфейс, отвечающий требованиям, описанным в ТЗ. Так как для приложения важна кроссплатформенность, то в качестве библиотеки для визуального отображения была выбрана PyQt6. Результатом данного этапа является визуальный каркас приложения, готового к внедрению логики, написанной на первом этапе работы.

# 3 этап. Объединение пользовательского интерфейса с математической частью

Срок выполнения этапа – 2 недели. Данный этап является заключительным. В результате его выполнения приложение является работоспособным и проверяется на соответствие с заявленным ТЗ.

Реализация математических функций, используемых для анализа звена.

# Построение графика переходного процесса.

Переходный процесс – зависимости сигнала на выходе системы, характеризованной какой-то передаточной функцией, при подаче на её вход ступенчатой функции(1(t)).

В ТАУ передаточная функция показ

ывает, как сигнал на выходе звена изменяется относительно сигнала на входе. Передаточная функция записывается как показано в формуле 2.

Для перехода от s-оператора к времени используется обратное преобразование Лапласа, которое представлено в формуле 3.

Данное вычисление переходного процесса является сложным в реализации, поэтому было принято решение использовать сторонние библиотеки, которые выполняют преобразование внутри себя.

Для построения переходного процесса была использована библиотека python-control, которая позволяет моделировать звенья с заданной передаточной функцией. Реализация данного модуля представлена на листинге 1.

Листинг 1

from control import \*  
from control.matlab import step  
  
from UI import mplwidget  
  
  
def draw\_transfer\_process(canvas: mplwidget, nm, dm):  
 canvas.axes.cla()  
 num = nm  
 den = dm  
 w = tf(num, den)  
 x, y = step\_response(w)  
 canvas.axes.grid()  
 canvas.axes.plot(x, y)  
 canvas.draw()

Таким образом был реализован метод, который получает на вход коэффициенты числителя и знаменателя передаточной функции и график, на котором необходимо построить данную зависимость. По заданным коэффициентам формируется передаточная функция, и далее моделируется её реакция на ступенчатое воздействие.

# Построение корневой картины.

Корневая картина для звена в ТАУ является изображением корней числителя и знаменателя на комплексной плоскости. Принято полюса знаменателя обозначать крестиками, а полюса числителя – нолями.

Таким образом, для получения корневой картины было необходимо найти корни полинома числителя и знаменателя и обозначить их на комплексной плоскости. Для нахождения корней полиномов была использована библиотека NumPy.

Для выполнения данной задачи была написана программа, представленная на листинге 2.

Листинг 2

import numpy as np  
from UI import mplwidget  
  
  
def draw\_root\_picture(canvas:mplwidget, coeffsNM, coeffsDN):  
 canvas.axes.cla()  
 canvas.axes.grid()  
 draw\_roots(canvas, coeffsNM, 'o')  
 draw\_roots(canvas, coeffsDN, 'x')

Продолжение листинга 2

canvas.draw()  
  
  
def draw\_roots(canvas:mplwidget, coeffs, marker):  
 tolerance = 0.000000001  
 roots = np.roots(coeffs)  
 no\_imag = True  
 no\_real = True  
 for i in range(len(roots)):  
 if abs(roots[i].imag) > tolerance:  
 no\_imag = False  
 if abs(roots[i].real) > tolerance:  
 no\_real = False  
 if len(roots) != 0:  
 if no\_real:  
 canvas.axes.set\_xlim([-1, 1])  
 if no\_imag:  
 canvas.axes.set\_ylim([-1, 1])  
  
 x = [ele.real for ele in roots]  
 y = [ele.imag for ele in roots]  
 canvas.axes.scatter(x, y, marker=marker)  
 canvas.axes.axhline()  
 canvas.axes.axvline()

Для правильного отображения корней, близких к нулю было добавлено допустимое отклонение.

# Построение частотных характеристик.

При построении частотных характеристик специалисты по ТАУ используют частоту в логарифмическом масштабе, чтобы показать весь обширный диапазон рабочих частот звена на одном графике.

Для построения ЧХ была так же использована библиотека python-control, которая моделировала работу звена и получала списки значений амплитуды и фазового сдвига на данной частоте. Программа, выполняющая данные вычисления представлена на листинге 3.

Листинг 3

from control import \*  
from numpy import rad2deg  
from UI import mplwidget  
  
def draw\_characteristics(mag\_canvas: mplwidget, phase\_canvas: mplwidget, nm, dm):  
 mag\_canvas.axes.cla()  
 phase\_canvas.axes.cla()  
 mag\_canvas.axes.grid()  
 phase\_canvas.axes.grid()  
  
 w = tf(nm, dm)  
 mag, phase, omega = bode(w, dB=True)  
 for i in range(len(phase)):

Продолжение листинга 3

phase[i] = rad2deg(phase[i])  
  
 mag\_canvas.axes.set\_xscale('log')  
 mag\_canvas.axes.plot(omega, mag2db(mag))  
  
 phase\_canvas.axes.set\_xscale('log')  
 phase\_canvas.axes.plot(omega, phase)  
 mag\_canvas.draw()  
 phase\_canvas.draw()

# Построение весовой характеристики.

Также в ходе выполнения работы было принято решение дополнительно реализовать возможность построения весовой характеристики звена.

Весовая характеристика – зависимости сигнала на выходе системы при подаче на её вход единичного импульса.

Вычисление данной зависимости схоже с вычислением переходного процесса. Разница в том, что в данном случае на вход системы вместо ступенчатой функции подается единичный импульс.

Данная задача была так же решена с помощью библиотеки python-control. Код программы представлен на листинге 4.

Листинг 4

from control.matlab import \*  
  
from UI import mplwidget  
  
  
def draw\_weight(canvas: mplwidget, nm, dm):  
 canvas.axes.cla()  
 num = nm  
 den = dm  
 print(num)  
 print(den)  
 w = tf(num, den)  
  
 y, x = impulse(w)  
  
 canvas.axes.grid()  
 canvas.axes.plot(x, y)  
 canvas.draw()

Разработка пользовательского интерфейса

При выполнении работы мы использовали средства разработки Qt Designer. Qt Designer – [кроссплатформенная](https://ru.wikipedia.org/wiki/%D0%9A%D1%80%D0%BE%D1%81%D1%81%D0%BF%D0%BB%D0%B0%D1%82%D1%84%D0%BE%D1%80%D0%BC%D0%B5%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5) [свободная](https://ru.wikipedia.org/wiki/%D0%A1%D0%B2%D0%BE%D0%B1%D0%BE%D0%B4%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5) среда для разработки [графических интерфейсов](https://ru.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D1%84%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%B8%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8F) (GUI) для программ, использующих библиотеку [Qt](https://ru.wikipedia.org/wiki/Qt" \o "Qt). Разработка интерфейса происходила следующим образом. На рис.2 слева находится панель инструментов, которые предоставляет приложение. В центре пространство для расстановки всех элементов и проектирования приложения. Справа – редактор свойств элементов.

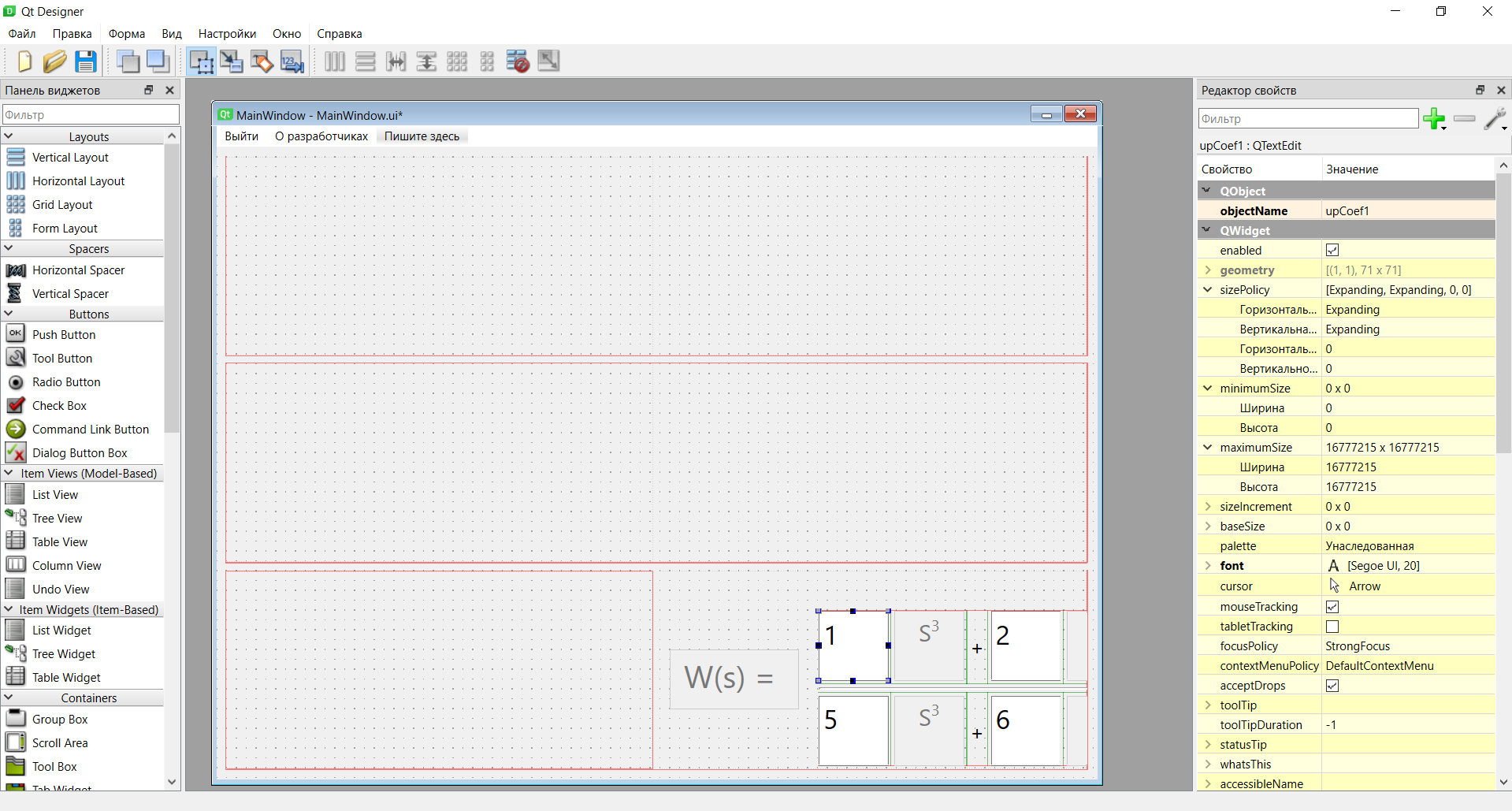


Рисунок 2. Разработка интерфейса

В результате мы получили следующий интерфейс (рис. 3):

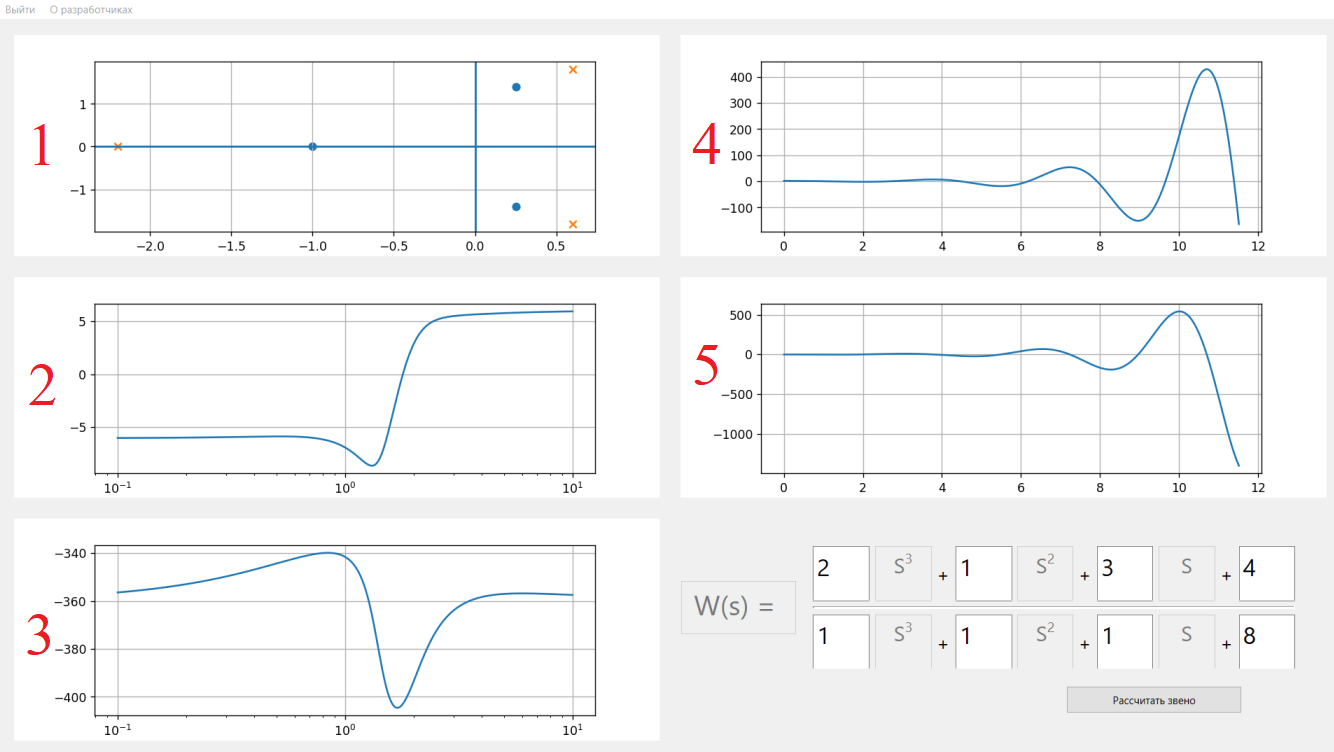


Рисунок 3. Интерфейс приложения

В правом нижнем пространстве пользователю предоставляется возможность ввести звено, там же и находится кнопка «Рассчитать звено». По нажатию этой кнопки появляются 5 графиков:

1. Корневая картина звена
2. Логарифмическая амплитудная характеристика
3. Логарифмическая частотная характеристика
4. Переходная характеристика
5. Весовая характеристика